REMARKS

Claims 1, 17, 29, 40 and 41 have been amended. Claims 1-41 remain pending in the application. Reconsideration is respectfully requested in light of the following remarks.

Section 102(b) Rejections:

The Examiner rejected claims 1, 15-17, 27-29, 39 and 41 under 35 U.S.C. § 102(b) as being anticipated by Tran (U.S. Patent 5,864,689) (also referencing "Intel Architecture Software Developer's Manual (hereinafter "Manual"), and claim 40 as being anticipated by Carbine et al. (U.S. Patent 5,630,083) (hereinafter "Carbine"). Applicants respectfully traverse this rejection for at least the following reasons.

Regarding claim 1, Tran fails to teach *wherein in response to receiving a microcoded instruction, the dispatch unit is configured to: replace the microcoded instruction in the instruction stream with a microcode subroutine call operation.*

The x86 CALL instruction is not a microcoded instruction, as would be easily understood by anyone of ordinary skill in the art, and Tran does not teach that the CALL instruction itself is a microcoded instruction (i.e., one executed in microcode). On page 11 of the Decision on Appeal, the Board states, "Our analysis begins with construing the claimed 'microcoded instruction' and we shall give this term its broadest reasonable construction in light of the specification as it would be interpreted by one of ordinary skill in the art… Appellants' Specification simply discloses that microcoded instructions are handled by a microcode unit and are often long running complex instructions (FF 1). Thus, we find that the claimed 'microcoded instructions' reasonably includes any complex instruction that can be handled by a microcode unit"

Applicants have amended the independent claims to address the Board's concerns. Applicants note that the quoted passage of Applicants' specification does not

constitute a definition of the term "microcoded instructions", but merely a description of some of the characteristics of microcoded instructions. Other passages of Applicants' specification (including paragraphs 4, 5, 47, 48, and 117) describe other characteristics of microcoded instructions, and describe differences between microcoded instructions and non-microcoded instructions. Applicants assert that in light of these additional passages, the above claim amendments, and the teachings of Tran, the x86 CALL instruction could not reasonably be considered a microcoded instruction implemented by a plurality of microcode operations. The CALL instruction in Tran is not described as itself being implemented by a plurality of microcode operations. Instead, the CALL instruction is a simple branch instruction (see, e.g., Tran claim 5, and column 4, lines 44 – 54, in which a standard subroutine call and return are described.) Tran teaches that if a branch type opcode (including a CALL opcode, for example) is detected, and the target address of the branch is within a particular range of addresses, the target address may be routed to microcode unit 45. In other words, Tran does not describe that the CALL instruction itself is implemented by a plurality of microcode operations.

In addition, Tran fails to disclose that in response to receiving a microcoded instruction, the dispatch unit is configured to replace the microcoded instruction in the instruction stream with a microcode subroutine call operation. Nothing in Tran describes the x86 CALL instruction being replaced by a subroutine call operation in the instruction stream. In fact, this would be illogical, since the x86 CALL instruction is itself a subroutine call operation.

Further regarding claim 1, Tran fails to teach that the dispatch unit is configured to *dispatch to the scheduler the microcode subroutine call operation, wherein the microcode subroutine call operation includes a tag identifying a microcode subroutine that is associated with the microcoded instruction and that comprises a plurality of microcode operations executable to implement the microcoded instruction.*

Tran teaches that if a branch type opcode (including a CALL opcode, for example) is detected and the target address of the branch is within a particular range of

addresses, the target address may be routed to microcode unit 45. This clearly does not teach the above-referenced limitation of Applicants' claim. For example, routing a target address of an instruction to a microcode unit clearly is not the same as dispatching to a scheduler a subroutine call operation that replaces the instruction in the instruction stream. Tran teaches no such replacement.

If Tran taught dispatching the xCALL instruction itself to the scheduler of an execution unit 38, this would be in direct contrast to the limitations of claim 1, in which *in response to receiving a microcoded instruction, the dispatch unit is configured to dispatch to the scheduler the microcode subroutine call operation* (i.e. the microcode subroutine call operation that replaced the microcoded instruction in the instruction stream). Applicants' claim clearly indicates that the received microcoded instruction and the dispatched microcode subroutine call operation are not the same instruction/operation. In response to receiving the first (the microcoded instruction), the second (the microcode subroutine call operation that replaces it) is dispatched to the scheduler. This replacement is clearly not taught by Tran.

Furthermore, Tran does not teach a microcode subroutine identified by the microcode subroutine call operation that *comprises a plurality of microcode operations executable to implement the microcoded instruction*. Tran describes that when the CALL instruction (which is a non-microcoded instruction) has a target address in a particular range, the target address is routed to the microcode unit, where it identifies the microcode routine to be executed. In Tran, the operations of the microcode routine called by the CALL instruction are dispatched to the execution units to perform a DSP function, not the CALL. These operations clearly are not *a plurality of microcode operations executable to implement the microcoded instruction*, as required by Applicants' claim (i.e. to perform the CALL instruction itself, using the Examiner's interpretation). For example, see column 4, lines 15-30, which states, in part:

> Microcode unit 45 fetches instructions that correspond to a routine which performs the indicated DSP function from the read-only memory…. Instructions fetched by microcode unit 45 are conveyed to execute units 38 and load/store unit 40.

Tran explicitly teaches that when a CALL instruction (which is not a microcoded instruction implemented by a plurality of microcode operations) having a target address in a particular range is detected, operations for executing a DSP function (not for executing the CALL instruction itself) are fetched beginning at the target address. Therefore, the use of a branch-type instruction to invoke a DSP function stored in microcode, as taught by Tran, clearly does not teach the limitations of Applicants' claim 1, as amended.

For at least the reasons above, the rejection of claim 1 is not supported by the cited art and removal thereof is respectfully requested.

Claim 17 includes limitations similar to claim 1, and so the arguments presented above apply with equal force to this claim, as well.

Regarding claim 29, contrary to the Examiner's assertion, Tran fails to teach or suggest *detecting a microcoded instruction within the stream of instructions, wherein the microcoded instruction immediately precedes an other instruction in program order, in response to said detecting, replacing the microcoded instruction in the instruction stream with a microcode subroutine call operation, wherein the microcode subroutine call operation identifies a microcode subroutine associated with the microcoded instruction*; and *dispatching the microcode subroutine call operation*.

As discussed in detail above in remarks directed to claim 1, Tran clearly does not disclose receiving (or detecting) a microcoded instruction in a stream of instructions, replacing it in the stream of instructions with a microcode subroutine call operation, or dispatching that microcode subroutine call operation. First, the x86 CALL instruction cited by the Examiner cannot be considered a microcoded instruction implemented by a plurality of microcode operations. In addition, Tran does not teach replacing the CALL instruction with a microcode subroutine call operation. Instead, Tran teaches the use of a

branch-type instruction (e.g., an x86 CALL instruction) to invoke a DSP function stored in microcode.

In addition, Tran does not disclose *executing a plurality of operations included in the microcode subroutine, wherein the plurality of operations includes a plurality of microcode operations executable to implement the microcoded instruction*. As described above in remarks directed to claim 1, Tran teaches that the operations of a microcode subroutine identified by the target address of an x86 CALL instruction perform a DSP function. They are not executable to implement the CALL instruction itself, as would be required by Applicants' claim, using the Examiner's interpretation.

Finally, Tran does not disclose that the microcode subroutine call operation *pushes an address of the other instruction onto a stack* and that *the plurality of operations included in the microcode subroutine includes a return operation, and wherein execution of the return operation pops the address from the stack*. The Examiner cites column 4, lines 42-54 as teaching that the microcode subroutine call operation pushes an address of the other instruction onto a stack, that the microcode subroutine includes a return operation, and that execution of the return operation pops the address from the stack (x86 Call instruction and x86 Return instruction). However, this citation has nothing to do with pushing or popping an instruction onto or off of a stack as part of a microcode subroutine call operation that replaces a microcoded instruction in the instruction stream and that is dispatched in response to detecting the microcoded instruction, as required by claim 29. Instead it describes that source-code level subroutine call instructions (e.g., the CALL instruction of the x86 instruction set) having target addresses within a particular range of addresses are indicative of DSP functions. In this example, these DSP functions are executed by branching to a subroutine explicitly programmed using the CALL and RET instructions of the x86 instruction set. Therefore, Tran cannot be said to anticipate claim 29

For at least the reasons above, the rejection of claim 29 is not supported by the cited art and removal thereof is respectfully requested.

Claim 41 includes limitations similar to claim 29, and so the arguments presented above apply with equal force to this claim, as well.

Regarding claim 40, Carbine fails to teach or suggest *dispatching one or more operations included in a first microcode subroutine and one or more operations included in a second microcode subroutine executing concurrently with the first microcode subroutine, wherein said dispatching the one or more operations in the first microcode subroutine comprises performing register name replacements using replacement register names stored in a first alias table element and wherein said dispatching the one or more operations in the second microcode subroutine comprises performing register name replacements using replacement register names stored in a second alias table element*. The Examiner cites column 12, lines 35-56, as teaching generic microcode routines, and asserts that each of these generic microcode routines "has micro-alias registers" to replace register names within the routine. The Examiner seems to be implying that these generic routines, including what he interprets as separate micro-alias registers, teach dispatching operations from both a first and second microcode subroutine executing concurrently with the first microcode subroutine, and the use of both a first and second alias table element in dispatching these two subroutines. However, there is nothing in this citation or elsewhere in Carbine that teaches multiple alias table elements or multiple micro-alias registers being used at the same time. Instead, Carbine describes a single micro-alias register (having multiple fields, but not multiple entries), which is loaded with different data dependent on which of four CUOPs is selected by multiplexer 560. There is also nothing in this citation that teaches dispatching multiple microcode subroutines, or concurrently executing microcode subroutines as required by claim 40. Instead, Carbine describes microcode sequencing unit 534 issuing multiple CUOPs for one microcode flow (associated with a single entry point) at a time (see, e.g., column 11, lines 11-13).

The Examiner previously submitted that Carbine specifically refers to a plurality of micro-alias registers at column 12, lines 36-39. The Examiner also previously

submitted that Carbine states that registers are not hard-coded into any routine, "thereby indicating multiple generic microcode routines are used. Therefore, Carbine has taught dispatching multiple generic microcode routines." Applicants assert, however, that even if multiple generic routines may exist in the system of Carbine, there is nothing that teaches dispatching two such routines, or executing them concurrently. While the system of Carbine is directed toward parallel decoding of multiple instructions (see Abstract) there is nothing in Carbine that discloses dispatching operations from two different microcode subroutines, as required in Claim 40. Applicants assert that it is clearly not necessary that a system support dispatching operations from two different routines, or executing them concurrently, even if they may be decoded in parallel, and the Examiner has not cited anything in Carbine that discloses this limitation of Applicants' claim. Applicants also note that in Carbine's Fig. 5, element 580 refers to "macro-alias registers" not "micro-alias registers."

Since Carbine fails to teach or suggest dispatching operations from a first and second microcode subroutine executing concurrently, and using a first and second alias table element in dispatching the first and second microcode subroutines, respectively, Carbine cannot be said to anticipate claim 40.

For at least the reasons above, the rejection of claim 40 is not supported by the cited art and removal thereof is respectfully requested.

Applicants also assert that numerous ones of the dependent claims recite further distinctions over the cited art. However, since the rejection has been shown to be unsupported for the independent claims, a further discussion of the dependent claims is not necessary at this time.

Section 103(a) Rejections:

The Examiner rejected claims 2-6, 18-22 and 30-34 under 35 U.S.C. § 103(a) as being unpatentable over Tran in view of Carbine, claims 7, 8, 23, 24, 35 and 36 as being

unpatentable over Tran and Carbine and further in view of Rotenberg, et al. ("Trace Cache: a Low Latency Approach to High Bandwidth Instruction Fetching") (hereinafter "Rotenberg"), claims 9, 10, 25 and 37 as being unpatentable over Tran in view of Kling (U.S. Publication 2004/0049657), and claims 11-14, 26 and 38 as being unpatentable over Tran and Kling and further in view of Harris (U.S. Patent 6,260,138). Applicants respectfully traverse these rejections for at least the reasons given above for the independent claims.

In regard to the rejections under both § 102(b) and § 103(a), Applicants assert that numerous ones of the dependent claims recite further distinctions over the cited art. Applicants traverse the rejection of these claims for at least the reasons given above in regard to the claims from which they depend. However, since the rejections have been shown to be unsupported for the independent claims, a further discussion of the dependent claims is not necessary at this time. Applicants reserve the right to present additional arguments.

CONCLUSION

Applicants submit the application is in condition for allowance, and prompt notice to that effect is respectfully requested.

If any fees are due, the Commissioner is authorized to charge said fees to Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C. Deposit Account No. 501505/5500-81600/RCK.

Respectfully submitted,

/Robert C. Kowert/
Robert C. Kowert, Reg. #39,255
ATTORNEY FOR APPLICANT(S)

Meyertons, Hood, Kivlin, Kowert, & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
Phone: (512) 853-8850

Date: August 31, 2009